LA-UR-16-25273

Weak Scaling of Schur-Complement Domain Decomposition for Kinetic Transport Sweeps

Student: Kevin Procopio[†], Mentor: Kris Garrett[§]

[†]*University of Pennsylvania* [§]*Los Alamos National Laboratory*

Abstract

Neutral particle transport, which includes the transport of neutrons, photons, and neutrinos, is an important phenomenon in determining the behavior of nuclear reactor cores and astrophysical processes. One deterministic method of simulating the transport of neutral particles is discrete-ordinates, which involves solving the radiative transfer equation (RTE) for a finite number of discrete solid angles. Parallelization of discrete-ordinates methods for use on supercomputing clusters can be difficult, especially for 3D, unstructured, tetrahedral meshes. In this project, the weak-scaling of various parallelization schemes for 3D discrete-ordinates methods were compared using Tycho2, a discontinuous Galerkin linear elements code developed at Los Alamos National Lab. Parallelization was done in conjunction with three algorithms: a parallel sweep in the direction of transport, Parallel Block Jacobi, and a Schur-Complement parallel Krylov subspace method. All three algorithms were run on the Cray XC30 system for various numbers of energy groups, numbers of solid angles, and domain sizes. MPI and OpenMP were used to implement the parallel algorithms.

Background

The behavior of neutral particles, such as neutrons, photons, and neutrinos, is important in the analysis of nuclear reactors and astrophysical phenomena. Kinetic transport of neutral particles is governed by the Radiative Transfer Equation

$$\Omega \cdot \nabla_x \Psi(x,\Omega,E) + \sigma_t \Psi(x,\Omega,E) = \frac{\sigma_s}{4\pi}\int_{\mathbb{S}^2} \Psi(x,\Omega',E)d\Omega' + Q(x,\Omega,E) \tag{1}$$

which balances material for a given $x \in D \subset \mathbb{R}^3$, $\Omega \in \mathbb{S}^2$, $E \in \mathbb{R}^{\geq 0}$. The function Ψ is the unknown, Q is a known source, and σ_t and σ_s are the total and scattering cross-sections (with $\sigma_t \geq \sigma_s$). Though the energy groups are not coupled, it is typical to discretize E into energy groups. Furthermore, discrete ordinates (S_N) methods can be used to discretize angle based on the quadrature of the sphere. Using discretization in E and Chebyshev-Legendre quadrature on the sphere for discrete ordinates, Equation (1) becomes

$$\Omega_q \cdot \nabla_x \Psi_{qg}(x) + \sigma_t \Psi_{qg}(x) = \frac{\sigma_s}{4\pi}\sum_{q'=0}^{2N^2-1} w_{q'}\Psi_{q'g}(x) + Q_{qg}(x) \tag{2}$$

where g is the index of the energy groups, q is the quadrature index, and the nodes and weights of the quadrature are Ω_q and w_q.

To deal with the integral on the RHS of equation (2), the data from the integral can be lagged. This is called source iteration

$$\Omega \cdot \nabla_x \Psi^{k+1} + \sigma_t \Psi^{k+1} = \frac{\sigma_s}{4\pi}\int_{\mathbb{S}^2} \Psi^k d\Omega' + Q \tag{3}$$

Since source iteration can be used to determine the RHS of equation (2), the scattering term can be absorbed in the source term to form a simplified equation

$$\Omega_q \cdot \nabla_x \Psi_{qg}(x) + \sigma_t \Psi_{qg}(x) = Q_{qg}(x) \tag{4}$$

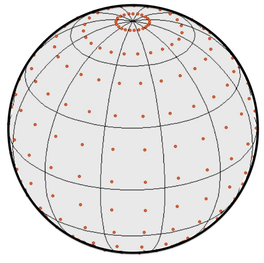


Figure 1: Chebyshev-Legendre quadrature

Multiplying (4) by a test function b, integrating, and then integrating by parts gives

$$-\int_{C_i} \Omega\cdot\nabla_x b_j \Psi dx + \sum_k \int_{F_{ik}} (\Omega\cdot\nu_k) b_j \hat{\Psi}^{(k)} dA + \int_{C_i} \sigma_t \Psi b_j dx = \int_{C_i} Q b_j dx. \tag{5}$$

Here, k indexes over the faces F_{ik} of cell i, and ν_k is the outward normal to face F_{ik}. Discretizations in angle and energy group are implicit.

Tycho2

Tycho2 is an open-source, discontinuous Galerkin linear elements code that solves the RTE using discrete ordinates on 3D, unstructured tetrahedral meshes. It can solve the RTE in parallel using several different algorithms. Note that since the elements are discontinuous, Ψ is discontinuous at the faces, and thus $\hat{\Psi}$ must be defined by the numerical method. Tycho2 uses the upwind flux

$$\hat{\psi}^{(k)} = \begin{cases} \Psi|_{F_{ik}} \text{ from } C_i, & \text{if } \Omega\cdot\nu_k > 0 \\ \Psi|_{F_{ik}} \text{ from adjacent cell}, & \text{if } \Omega\cdot\nu_k < 0 \end{cases} \tag{6}$$

Algorithms

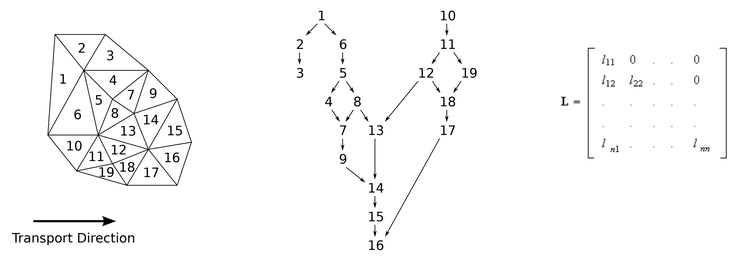


Figure 2: (Left) 2D unstructured mesh and chosen transport direction for sweep method. (Center) Directed acyclic graph produced as a result of the mesh. (Right) Performing the sweep is equivalent to inverting a sparse lower triangular matrix.

Sweeps:

As depicted in Figure 2, the method of sweeps involves selecting a transport direction and generating a list of priorities for computing each of the cells. This is the same as forming a directed acyclic graph (DAG) of cell dependencies, or inverting a sparse lower triangular matrix. Note that Figure 2 depicts a 2D example, but Tycho2 uses a 3D tetrahedral mesh.

Optimal scheduling for a variable number of processors and a general DAG is NP-Complete. Even if the number of processors is known, the complexity of the optimal scheduling problem has unknown complexity for more than two processors. However, several heuristics exist. Tycho2 supports several different heuristics for parallel sweeping.

Parallel Block Jacobi:

With the Parallel Block Jacobi (PBJ) algorithm, each MPI rank has a guess for the boundary data and sweeps are performed on each partition of the mesh independently of the other partitions. Then, boundary values are communicated between all of the ranks and sweeps are repeated. This process continues until Ψ has converged to a given tolerance. Parallel Block Jacobi is equivalent to performing a fixed-point iteration on a diagonal block matrix, as depicted in Figure 3.

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{A}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{A}_n \end{bmatrix}$$

Schur-Complement Krylov:

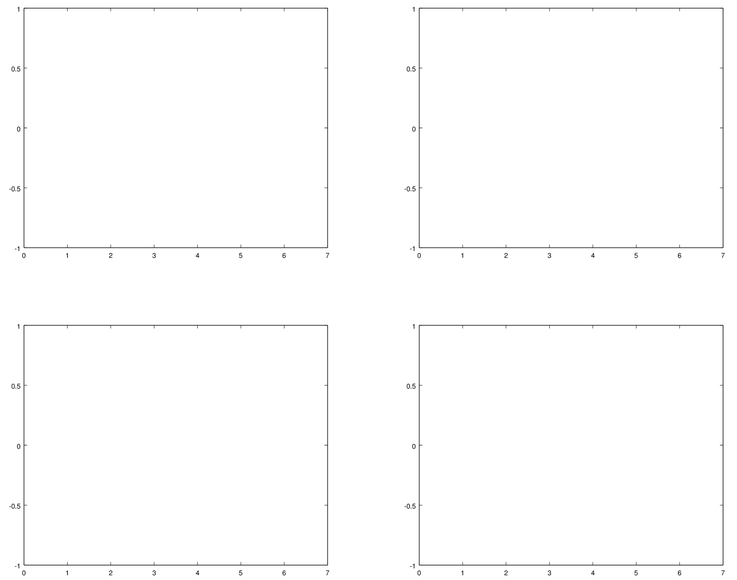
With the Schur-Complement Krylov algorithm, the mesh is partitioned into regions (similar to the PBJ partitioning) and a boundary partition. Each MPI rank iteratively solves the local boundary of one of the partitions using a Krylov solver. Communication occurs between each rank until the global boundary partition has converged to a given tolerance. Then, each rank performs a sweep in parallel to solve the internal values for each partition.

The Schur-Complement Krylov method is equivalent to rearranging the lower triangular matrix used in the sweep, taking its Schur Complement, solving the boundary, and then solving the other blocks in parallel. Figure 4 depicts the rearrangement of a lower triangular matrix into the appropriate form for the algorithm as well as the Schur-Complement of said matrix.

$$\begin{bmatrix} A_{11} & 0 & \dots & A_{\Gamma 1} \\ 0 & A_{22} & \dots & A_{\Gamma 2} \\ \vdots & \vdots & \ddots & \vdots \\ A_{1\Gamma} & A_{2\Gamma} & \dots & A_{\Gamma\Gamma} \end{bmatrix} \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \vdots \\ \Psi_\Gamma \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_\Gamma \end{bmatrix} \Longrightarrow \begin{bmatrix} A_{11} & 0 & \dots & A_{\Gamma 1} \\ 0 & A_{22} & \dots & A_{\Gamma 2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & A_{\Gamma\Gamma} - \sum_i A_{\Gamma i} A_{ii} A_{i\Gamma} \end{bmatrix} \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \vdots \\ \Psi_\Gamma \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \\ \vdots \\ Q_\Gamma - \sum_i A_{\Gamma i} A_{ii} Q_i \end{bmatrix}$$

Results

THIRD COLUMN IS DRAFT WAITING ON RESULTS



Above are scaling plots....

Conclusion

We conclude that...

References

We cite...